GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

ADVANCED DRIVING LEARNING SYSTEM

A Project report of Capstone Project - 2

Submitted by

Sudhanshu Singh (1613107063)

in partial fulfillment for the award of the degree of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of
Dr. Varun Tiwari
Assistant Professor

APRIL / MAY- 2020

TABLE OF CONTENTS

Chapter 1. Abstract

One of the main challenges for advanced driver assistance systems (ADAS) is the environment perception problem. One factor that makes ADAS hard to implement is the large amount of different conditions that have to be taken care of. The main sources for condition diversity are lane and road appearance, image clarity issues and poor visibility conditions. A review of current lane detection algorithms has been carried out and based on that a lane detection algorithm has been developed and implemented on a mixed criticality platform. The final lane detection algorithms consists of preprocessing steps where the image is converted to gray scale and everything except the region of interest (ROI) is cut away. OpenCV, a library for image processing has been utilized for edge detection and hough transform. An algorithm for error calculations is developed which compares the center and direction of the lane with the actual vehicle position and direction during real experiments. The lane detection system is implemented on a Raspberry Pi which communicates with a mixed criticality platform. The demonstrator vehicle can achieve a measured speed of 3.5 m/s with reliable lane keeping using the developed algorithm. The advantage of the approach is that even without lane markings, lanes still can be found only with the information of their boundaries. However, problems also exist. For example, edge detection is time-consuming because of a great amount of calculation so it is hard to be a real time system. Furthermore, the problem of noises such as shadows of obstacles and vehicles may cause errors of the detection. In addition, edge detection requires features of edges on a lane's both sides. If obstacles such as the preceding vehicles occlude one side, errors may arise.

Chapter 2. Introduction

Using the machine learning algorithms and python it will predict the width of the road .It will also tell the distance from the roadside and also guide for overtaking the vehicles.

The trained machine become the powerful feature in the driverless car as the vehicle get proper assistance in the overtaking and parking also .It can also be used in the cruise control of the car make the more impact of the car and easy in controlling the car.

Active safety is currently a key topic in the automotive industry, which fosters the development of Autonomous Vehicle functions. Various advanced driver assistance systems (ADAS) and active safety systems have the potential to improve road safety, driver comfort, fuel economy, and traffic flow by assisting the driver during different driving conditions. It is estimated that human error is a contributing factor in more than 90% of all accidents. In order to save the human lives caused by road accidents, it is hence of interest to develop such systems using modeling and simulation tools which is quick and more efficient as compared to the real driving testing. Overtaking is one of the most complex maneuvers with the high risk of collision (75% human error) so the automation of this maneuver still remains one of the toughest challenges in the development of autonomous vehicles.

Since overtaking is one of the complex maneuvers and so many factors affect it, the automation of this maneuver has been considered to be one of the toughest challenges in the development of autonomous vehicles. Overtaking involves a great interaction between both longitudinal (throttle and brake) and lateral (steering) actuators. Nowadays, in the field of driver assistance systems and automated driving, development approaches for lateral maneuver control are the very big challenge.

LITERATURE SURVEY

Machine Learning:

At a very high level, machine learning is the process of teaching a computer system how to make accurate predictions when fed data. Those predictions could be answering whether a piece of fruit in a photo is a banana or an apple, spotting people crossing the road in front of a self-driving car, whether the use of the word *book* in a sentence relates to a paperback or a hotel reservation, whether an email is spam, or recognizing speech accurately enough to generate captions for a YouTube video. The key difference from traditional computer software is that a human developer hasn't written code that instructs the system how to tell the difference between the banana and the apple. Instead a machine-learning model has been taught how to reliably discriminate between the fruits by being trained on a large amount of data, in this instance likely a huge number of images labelled as containing a banana or an apple.

Open CV:

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies. Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many start-ups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

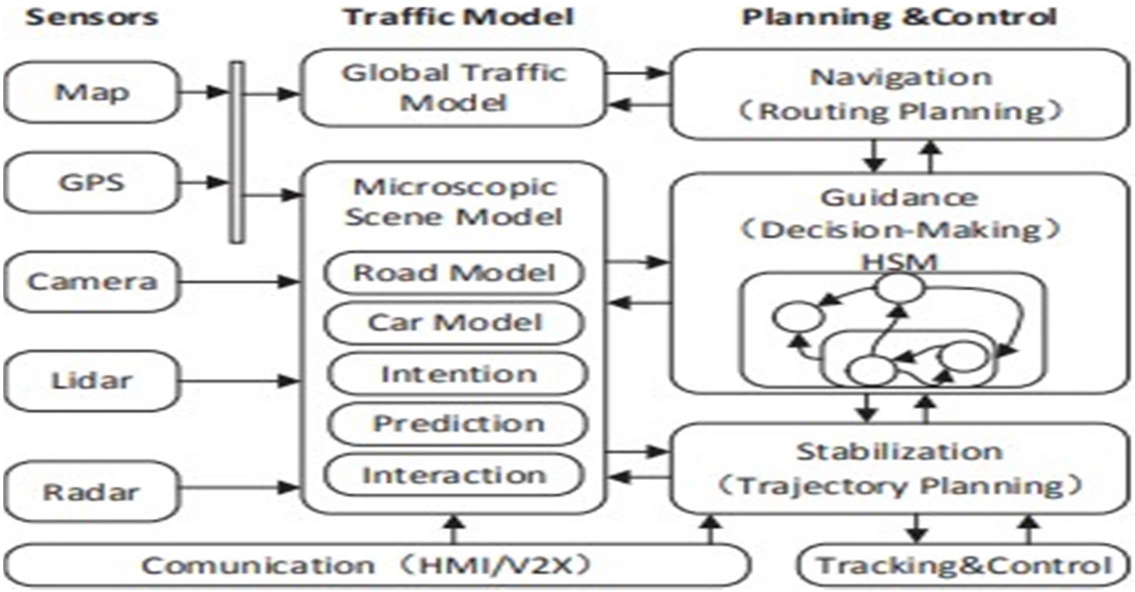
Advanced Driving Assistance Systems:

Advanced Driver Assistance Systems are intelligent systems that reside inside the vehicle and assist the main driver in a variety of ways. These systems may be used to provide vital information about traffic, closure and blockage of roads ahead, congestion levels, suggested routes to avoid congestion etc. These systems may also be used to judge the fatigue and distraction of the human driver and thus make precautionary alerts or to assess the driving performance and make suggestions regarding the same. These systems can take over the control from the human on assessing any threat; perform easy tasks (like cruise control) or difficult manoeuvres (like overtaking and parking). The greatest advantage of using the assistance systems is that they enable communication between different vehicles, vehicle infrastructure systems and transportation management centres. This enables exchange of information for better vision, localization, planning and decision making of the vehicles. Camera systems in ADAS can display what is behind or beside the vehicle, even at night on screen. They can also analyse the video content for automatic lane-departure warning systems and high/low-beam headlight control. An image sensor interface provides incoming video frames to a single- or dual-core architecture optimized with DSP extensions for image improvement filtering and edge or spot detection. Additional system requirements include an appropriate communication interface, an integrated DRAM interface for fast access to external memory and embedded flash for low system cost.

Classification:-

Camera systems in ADAS can display what is behind or beside the vehicle, even at night on screen. They can also analyse the video content for automatic lane-departure warning systems and high/low-beam headlight control. An image sensor interface provides incoming video frames to a single- or dual-core architecture optimized with DSP extensions for image improvement filtering and edge or spot detection. Additional system requirements include an appropriate communication interface, an integrated DRAM interface for fast access to external memory and embedded flash for low system cost. In machine learning, classification is a supervised learning concept which basically categorizes a set of data into classes. The most common classification problems are – speech recognition, face detection, handwriting recognition, document classification, etc. It can be either a binary classification problem or a multi-class problem too. There are a bunch of machine learning algorithms for classification in machine learning. Let us take a look at those classification algorithms in machine learning.

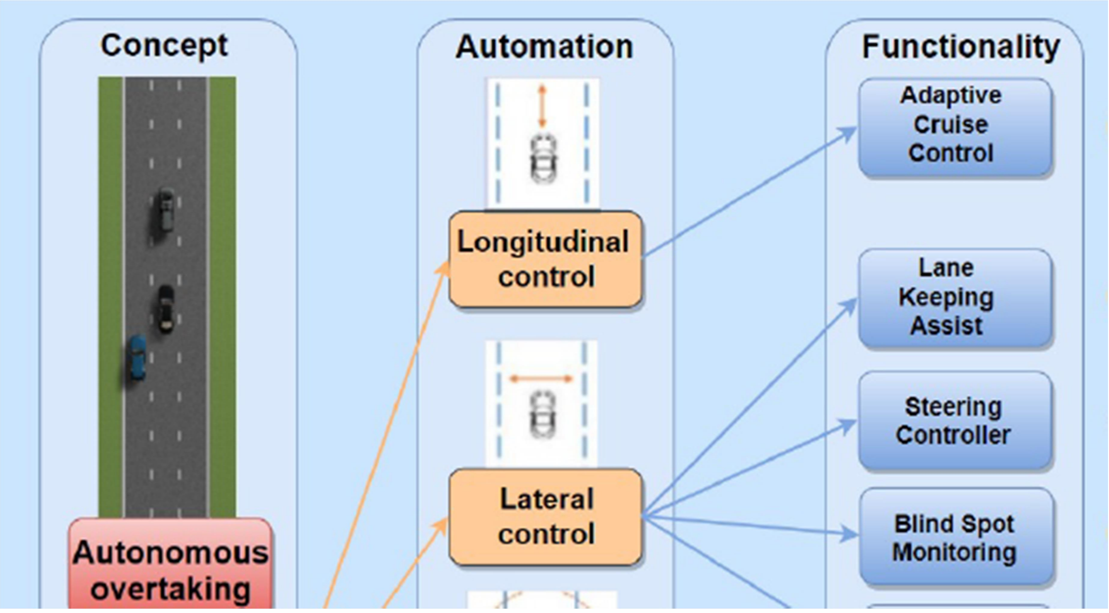
Chapter 3. Existing System



A tracking problem is usually not an initial value problem, and one needs to harvest the boundary conditions at the final time. Even if one assumes that, a solution to the boundary value problem exists, the nature of the trajectory cannot be guessed from the problem description if the system fails to have an analytic solution, as also pointed out. A numerical procedure is the norm in such cases and requires an assumption that the system remains reachable under the assumed set of initial values and sampling interval.

The possible solution for comfort oriented vehicle following with leading vehicle movement prediction treated as disturbance controller is presented. It is dealing with the execution of optimal speed trajectory planning is done in a way that modifying an optimal speed trajectory leads to the smallest deviation from the desired speed while the vehicle is moving. These approaches are treating the problem locally and partially and don't give an energy consumption based decision if a vehicle should overtake.

Chapter 4. Proposed System



This project presents an advanced lane detection technology to improve the efficiency and accuracy of real-time lane detection. The lane detection module is usually divided into two steps: image pre-processing and the establishment and matching of line lane detection model.

Figure 1 shows the overall diagram of our proposed system where lane detection blocks are the main contributions of this paper. The first step is to read the frames in the video stream. The second step is to enter the image pre-processing module. What is different from others is that in the pre-processing stage we not only process the image itself but also do colour feature extraction and edge feature extraction. In order to reduce the influence of noise in the process of motion and tracking, after extracting the colour features of the image, we need to use Gaussian filter to smooth the image. Then, the image is obtained by binary threshold processing and morphological closure. These are the pre-processing methods mentioned in this project.

In this project I will attempt to find lane lines from a dash cam video feed. Once we detect lane lines, we will mark them on the original video frame and play it back. This all will be done online and without any lag using OpenCV functions.

Our approach here would be to develop sequence of functions to detect lane lines. We will use a ‘sample’ image while writing this functions, and once we are able to detect lane lines on few ‘sample’ images successfully, we will club complete program into a function which can accept live feed image and return the same image frame with lane lines highlighted.

Using the machine learning algorithms and python it will predict the width of the road .It will also tell the distance from the roadside and also guide for overtaking the. It can also used in the cruise control of the car make the more impact of the car and easy in controlling the car. The trained machine will able to calculate and it will predict the width of the lane that will help to drive the car in its lane with the proper guidance.

Canny Edge Detector Algorithm -The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986. Canny also produced a computational theory of edge detection explaining why the technique works.

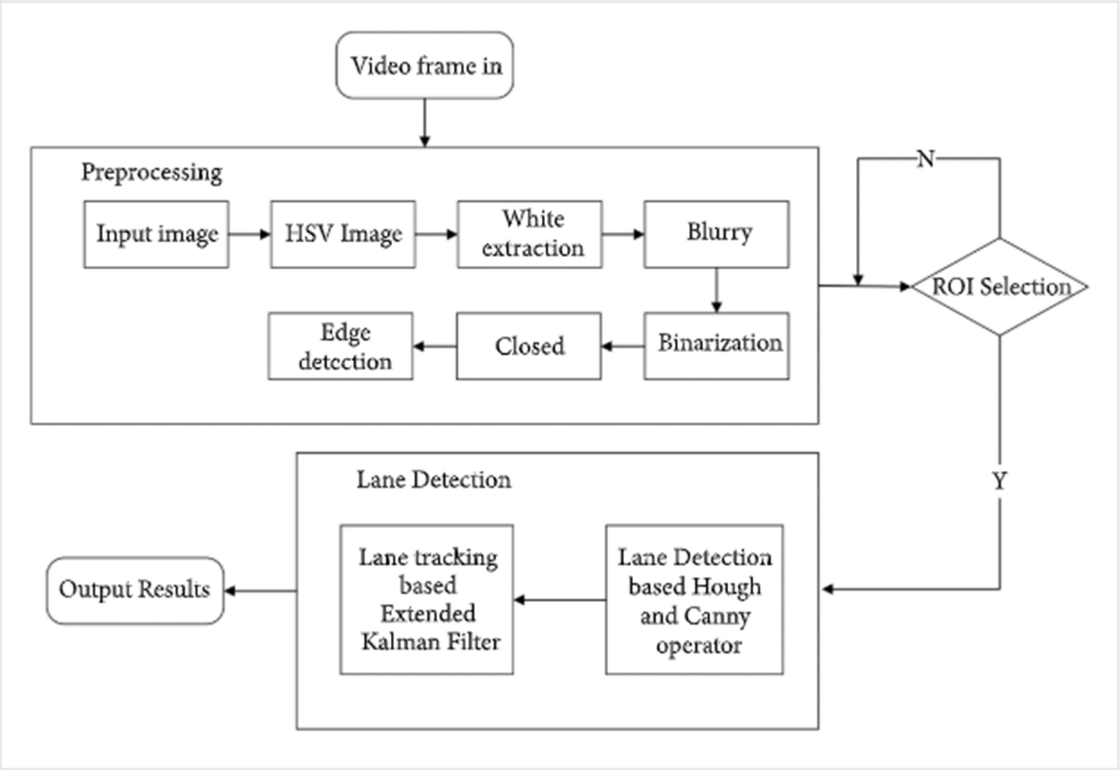
Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed.

- Detection of edge with low error rate, which means that the detection should accurately catch as many edges shown in the image as possible
- The edge point detected from the operator should accurately localize on the center of the edge.
- A given edge in the image should only be marked once, and where possible, image noise should not create false edges.

Process of Canny Edge Detection Algorithm

1. Apply Gaussian filter to smooth the image in order to remove the noise
2. Find the intensity gradients of the image
3. Apply non-maximum suppression to get rid of spurious response to edge detection
4. Apply double threshold to determine potential edges
5. Track edge by hysteresis: Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges

Block Diagram



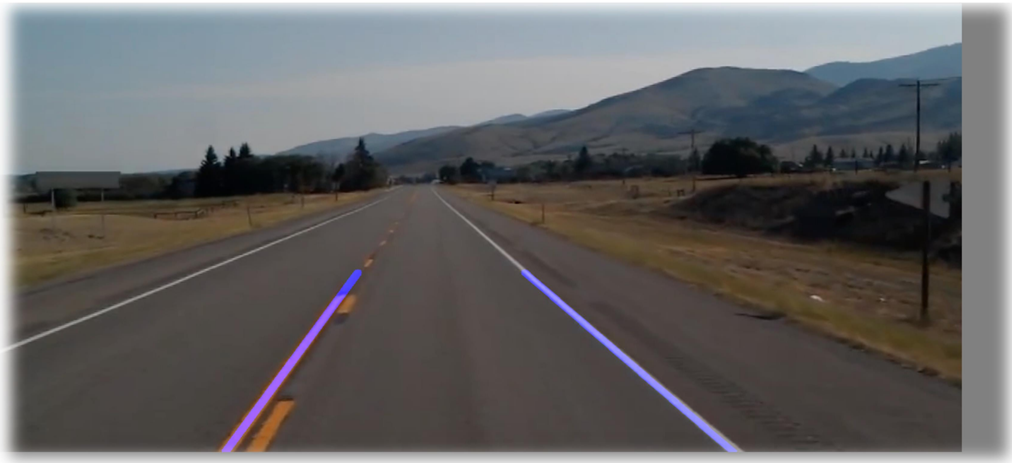
Chapter 5. Implementation

Program Code:

```
import cv2

import numpy as np

def make_coordinates(image,line_parameters):

    slope,intercept=line_parameters

    print(image.shape)

    y1=image.shape[0]

    y2=int(y1*(3/5))

    x1=int((y1-intercept)/slope)

    x2=int((y2-intercept)/slope)

    return np.array([x1,y1,x2,y2])

return



def average_slope_intercept(image,lines):

    left_fit=[]

    right_fit=[]
    for line in lines:
x1,y1,x2,y2=line.reshape(4)
        parameters=np.polyfit((x1,x2),(y1,y2),1)
        slope=parameters[0]
        intercept=parameters[1]
        if slope <0:
            left_fit.append((slope,intercept))
        else:
            right_fit.append((slope,intercept))
    left_fit_average=np.average(left_fit,axis=0)
    right_fit_average=np.average(right_fit,axis=0)
    left_line=make_coordinates(image,left_fit_average)
    right_line=make_coordinates(image,right_fit_average)
    return np.array([left_line,right_line])
def canny(image):
    gray=cv2.cvtColor(image,cv2.COLOR_RGB2GRAY)
    blur=cv2.GaussianBlur(gray,(5,5),0)
    canny=cv2.Canny(blur,50,150)
    return canny

def display_lines(image,lines):

    line_image=np.zeros_like(image)

    if lines is not None:

        for x1,y1,x2,y2 in lines:

            cv2.line(line_image,(x1,y1),(x2,y2),(255,0,0),10)

    return line_image

def region_of_interest(image):

    height=image.shape[0]

    polygons=np.array([

    [(200,height),(1100,height),(550,250)]

    ])

    mask=np.zeros_like(image)

    cv2.fillPoly(mask,polygons,255)

    masked_image=cv2.bitwise_and(image,mask)

    return masked_image

cap=cv2.VideoCapture("test2.mp4")

while(cap.isOpened()):
    _, frame=cap.read()

    canny_image=canny(frame)

       cropped_image=region_of_interest(canny_image)


lines=cv2.HoughLinesP(cropped_image,2,np.pi/180,100,np.array([]),minLineLength=40,maxLineGap=5)

    averaged_lines=average_slope_intercept(frame,lines)

    line_image=display_lines(frame,averaged_lines)

    combo_image=cv2.addWeighted(frame,0.8,line_image,1,1)

    cv2.imshow("result",combo_image)

    if cv2.waitKey(1)==ord('q'):

        break

cap.release()

cv2.destroyAllWindows
```

Chapter 6. Output / Screenshot

Input (Sample Image)



Output Screen

Chapter 7. Conclusion

The problem of autonomous highway overtaking was solved. The test protocol for highway overtaking assist was developed which was further used for the development of an automated driving system for the autonomous highway overtaking. The developed test protocol was validated analytically using mathematical equations and the automated driving system was tested virtually. The simulation results were found to be in accordance with the desired host vehicle behavior. The system drives the host vehicle through the selected use cases in a safe and efficient manner, while interacting with target vehicles operating in the traffic environment. The proposed autonomous highway overtaking system has the following characteristics:

- Safe, comfortable, and robust: The safety of the developed system was guaranteed by ensuring that the host vehicle remains outside the safe time gap during overtaking. Also, the host vehicle was able to execute the overtaking maneuver without exceeding the safe limits of longitudinal acceleration, heading angle, yaw rate, and lateral velocity, it was concluded that the developed fuzzy controller was suitable for application to the steering control even at a higher speed. Thus, the system is robust enough.
- Feasible and modular framework of the autonomous highway overtaking system The developed system is feasible since it uses the sensors and ADAS systems Also, the development of subsystems is performed independently which might be beneficial to the future optimization of the system under study and implementation into a real vehicle. The system can be said to be optimized for real-time implementation.

Chapter 8. References

[1] G. Hegeman, K. Brookhuis, S. Hoogendoorn, et al., “Opportunities of advanced driver assistance systems towards overtaking,” EJTIR, vol. 5, no. 4, pp. 281–296, 2005.

[2] F. Tillema, “PRAUTOCOL, Certification of drivers and cars in the field of autonomous driving (RAAK SME), Smart mobility, HAN Automotive Research,” 2017.

[3] A. Khodayari, F. Alimardani, and H. Sadati, “Modeling and intelligent control system design for overtaking maneuver in autonomous vehicles,” Iranian Journal of Mechanical Engineering Transactions of the ISME, 2012.

[4] “Automated vs. Autonomous Vehicles: Is There a Difference?.” https://www.autotrader.com/ car-news/automated-vs-autonomous-vehicles-there-difference-273139, 2018. [Online; accessed 25-04-2018].

[5] J. E. Stellet, M. R. Zofka, J. Schumacher, T. Schamm, F. Niewels, and J. M. Zöllner, “Testing of advanced driver assistance towards automated driving: A survey and taxonomy on existing approaches and open questions,” in Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on, pp. 1455–1462, IEEE, 2015.

[6] G. Hegeman, Assisted overtaking: An assessment of overtaking on two-lane rural roads. No. T2008/4, TRAIL Research School Delft, the Netherlands, 2008.